**CP2406 Assessment Task 1**

1. **User Stories**

User Story 1:

As a user, I want to know the number of vehicles driving on the road so that I can understand the traffic congestion.

Priority: High

Estimate: 3 hours

Test: change the number creating per vehicle

User Story 2:

As a user, I want to know what kind of vehicle is suitable for driving in the road.

Priority: Low

Estimate: 1 hour

Test: Chang different kind of vehicle in the different in the road

User Story 3:

As a user, I want to know the speed of different vehicles

Priority: Medium

Estimate: 2 hours

Test: change the speed of different vehicles

User Story 4:

As a user, I want to know the changes of traffic lights to understand the speed of the vehicle.

Priority: Medium

Estimate: 2 hours

Test: Change traffic lights to test vehicle speed

Users:

* Choose different vehicle type in the different road
* Choose different seep through traffic lights
* Change different speed with different vehicle type
* Choose a road based on traffic

1. **Problem specification**

A car simulator is to be implemented using road objects that interact with each other. There needs to be at least a road, traffic light and vehicle object with a Main class acting as the simulator. The program is to be coded in java.

1. **Problem decomposition**

The objects that will be used are a Road, Traffic Light, Car and various graphic objects. A Main class is used to have these objects interact with each other.

1. **Class Design**

**1）Traffic Light**

A Traffic Light object is used to tell a vehicle object (e.g. Car) passing the road to stop or go. Traffic Light keeps track of its position which is taken by the last Road length segment. The constructor will take a Road object as a given value and use its getter method to obtain the Road length and set the value as its position. The Traffic Light class keeps track of the Road it is on, position and its signal to go (Boolean where true equals go and false is stop).

The class also has a method that will be used to randomly change the Boolean signal value repeatedly in the simulator. It will be public as Main class repeatedly calls this method.

**2）Car**

Objects of the Car class have the role of driving on a road and through a traffic light when the traffic light is green but stopping when it is red. The member fields needed are; id width, height, color, pass\_ speed, location etc. These will be private since they won’t be used outside of itself. The methods needed are; getWidth (), getHeight (), getColor (), getTimeStopped (), getPassSpeeds (), etc. These methods are to be public since they will be used outside of itself. The Car class will extend the Simulation class. A Road object is needed for each of the other classes to be simulated. Each Road object stores its length in a member field and other objects (such as a Car object) uses a segment of that length to track its position on the Road. Length is private so that other objects cannot change its value. The only member field in this class is length.

**3）Bus**

The bus category will be a subcategory of cars, describing large road vehicles. It will inherit its properties and behavior from Car class

**4）Motorbike**

The motorcycle category will be a subcategory of Car, describing small road bikes. It will inherit Car's properties and behavior.

**5）Road**

Objects of the Road class have the role of having a car drive over them and having a traffic light at either the start of the road or the end of the road. The member fields needed are; road\_id, length, traffic\_light\_end, traffic\_light\_start. These member fields will be private as they are not being accessed outside of the class. The methods of this object that are needed are; getRoadID (), getLength (), getTrafficLightEnd (), getTrafficLightStart (), etc. These methods are to be public since they will be accessed in the Main class. The Road class will extend the Simulation class.

**6）Four\_way**

The Four\_way category will be a subcategory of Road, describing the type of road. It will inherit the properties and behavior of the Road class

**7）Three\_way**

The Three\_way category will be a subcategory of Road, describing the type of road. It will inherit the properties and behavior of the Road class

**8）Straight**

The Straight category will be a subcategory of Road, describing the type of road. It will inherit the properties and behavior of the Road class

**9）Main**

This class will have the main () method that will contain the simulation loop.

All the objects needed for the simulation will be created here; creating roads, a car and a traffic light. Locations for roads and their connection will also be set here.

**10）Simulation**

The Simulation class will be abstract and be used as a parent class. The member fields needed are; road\_list, traffic\_light\_list, car\_list, etc. These will be private as they will only be accessed within its own class and its child classes. The methods are; placeRoad(), placeTrafficLight(),placeVehicle(),etc. These will be public since they will be accessed outside of itself.

1. **Method Design:**

1）The method signature of Main is: main () and exit ().

-Main will be used to initialize the Main class.

-exit () will be used to animate the cars driving along the road and change the state of trafficlight.

-main () will be used to initialize a new instance of Simulation.

2）The method signatures of Simulation are: placeRoad (), placeTrafficLight (), placeVehicle () and runSim () are used to initialize Simulation and its subclasses. .

-runSim () is used to initialize Simulation and set the GUI.

-PlaceRoad () is used to draw all the contents of the road.

placeTrafficLigh () is used to draw all traffic ligh content.

placeVehicle () is used to draw the contents of the vehicle parking

3）Road's method signatures are Road (int, Boolean).

-Road (int, Boolean) is used to initialize the road object by setting the variable of the road object.

-traffic\_light\_end (Boolean) Used to display the road as vertical.

 traffic\_light\_start (Boolean) is used to display the road as level.

-getTrafficLightStart () is used to return the current X value.

-getTrafficLightStart () returns the current y value.

-getLength () returns the length of the road.

-getWidth () returns the width of the road.

-getRoadID ()) is used to draw the road according to the color and direction of the road.

4）TrafficLight's method signature will be TrafficLight (int).

-TrafficLight (int, Boolean) is used to initialize TrafficLight class objects by setting all variables.

-getTrafficLightID () is used to draw trafficLight according to its color and size.

-getOnRoad () returns the current state of trafficLight.

-getIsGreen (), GetIsOrange (), etc are used to return the color of trafficLight according to the status.

Vehicle's method signatures are Vehicle (int, int), move (), paintComponent (Graphics) and update (TrafficLight, Boolean, Road []).

5）Car's method signature is Car (int, Boolean) used to initialize all member fields with default values.

-move () will set the x and y values ​​based on the xDir and yDir variables.

-setBehind (int) is used to set behind according to the incoming int.

-setFront (int) is used to set the front according to the incoming int.

-getBehind () is used to return the current X value.

-getFront () is used to return the current y value.

-getLength () returns the length of the car.

-getWidth () returns the width of the car.

-getColor () is used to return the color of the car.

-getVhehicle () is used to return whether the car is level.

-getOnRoad () is used to return a random direction.

-update (TrafficLight, Boolean, Road []) is used to update the car's output on getOnRoad () based on the TrafficLight's x position, TrafficLight's status, whether the car is in a horizontal state, road variables, and the direction to travel.

-displayCars () is used to reset the position of the car when it reaches the end.

-setColor (Graphics) is used to draw the car according to the color and direction of the car.